VERUM

Why Aircraft Fly and Software Systems Don't

Robert Howe

Contents

Introduction

- Background
  - Degree in Physics from Manchester University in 1982
  - Options in Electronics & Computing
  - Options in Aerodynamics and Flight Mechanics
  - Pilot with +/- 900 hours flying experience
- Industry
  - 15 years software engineering experience
  - Founded Mountside Software Engineering in 1989
  - Focused on Business Development from 1995
  - Size: 55 people, Turnover €3.5 million
  - Sold Mountside 2001

The Success of Engineering

BBC NEWS UK EDITION

WATCH/LISTEN TO BBC NEWS

News Front Page
World
Africa
Americas
Asia-Pacific
Europe
Middle East
South Asia
UK
England
Northern Ireland
Scotland
Wales
Business
Politics
Health
Education
Science/Nature
Technology
Entertainment
Have Your Say
Magazine
In Pictures
Week at a Glance
Country Profiles
In Depth
Programmes

Last Updated: Tuesday, 25 May, 2004, 13:21 GMT 14:21 UK

E-mail this to a friend | Printable version

Mystery over the Paris airport collapse

By Kate Milner
BBC News Online

Structural engineers are baffled by what might have caused the roof collapse at Paris' Charles de Gaulle-Roissy airport, which killed four people.

The ultra-modern Terminal 2E, which opened 11 months ago, was built using steel, concrete and 36,000 sq m (118,116 sq ft) of reinforced glass.

It consists of two long tunnel-like buildings connected by a central passageway.

The part that collapsed was a 30-metre (98-foot) section of roof on the outer building, where passengers wait before boarding the aircraft.

The futuristic building was based on ideas learned in tunnel construction

The investigation into what went wrong will focus primarily on whether there was a fault in the design of the building, or whether short cuts had been taken when it was built.

The internationally-renowned French architect who designed the terminal, Paul Andreu, has described the structure as "bold" but "nothing revolutionary".

SEE ALSO:
- 'Fresh cracks' at Paris airport
  24 May 04 | Europe
- Profile: Paul Andreu
  24 May 04 | Europe
- France's showcase airport under spotlight
  23 May 04 | Europe
- Paris airport roof collapse: Your reaction
  24 May 04 | Have Your Say

RELATED INTERNET LINKS:
- Paris Charles de Gaulle airport
- Expedition Engineering

The BBC is not responsible for the content of external internet sites

TOP EUROPE STORIES NOW
- Putin under pressure over siege
- First witness for Milosevic due
- Unions may block Alitalia rescue
- 'McLibel' pair continue court bid

Charles de Gaul Airport

“The structure stood for a year and then spontaneously fell down without anything apparently changing. It's very odd”

“Buildings like that do not just collapse”, leading structural engineer Professor Chris Wise told BBC News Online.

"Something very unusual has happened," he said.

So could a mistake have slipped through?

“There are lots of checks and balances in a project like this”, Professor Wise said. "If you make a mistake with the design there's almost no chance of your design getting built. Somebody along the way would find it."

Unpredictability

- Society does not tolerate engineering failures
- Engineering does not tolerate unpredictability
- Consider the world of Aeronautical Engineering

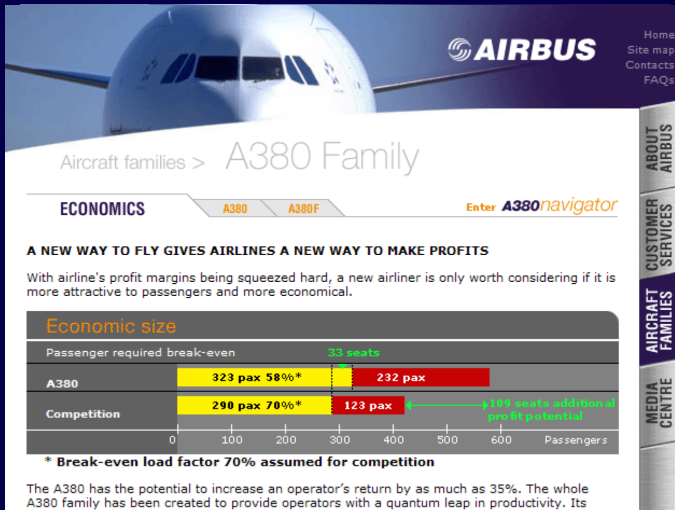
Contents

Airbus A380

A380 Economics

- Cost of an A380 +/- $240 Million
- Airbus need to sell +/- 300 to break even
  - $72 billion to break even
- Development started in 1999
- First deliveries scheduled for 2006
- Currently have orders for 139
- How did they manage to sell 139 airplanes of a type that has never flown?

Advanced Knowledge



Testing the A380

The Purpose of Testing

- Testing is **not** about finding out whether aircraft will fly.
  - It is tradition for Boeing's design team to always sit in on the inaugural flight of a new aircraft type
- Testing is not about removing Requirements, Architecture or Major Design Defects
  - It would be economic death for the A380 if it were
- Testing is largely about confirming what they already know.
- Confirming what they know is all about Predictability

Why do Aircraft Fly?

- Because, apart from everything else, Aircraft Designers and Aeronautical Engineers make routine, systematic use of mathematics to describe and control the complexity of their product.
- Aircraft Designers and Aeronautical Engineers are taught to make use of mathematics to describe complexity from the moment they enter higher education.
- The Aeronautical Engineering world makes use of the conventions, tools and mathematical techniques developed by Universities and taught to their students

Contents

Circuit Engineering[1]

- Born in 1938 as a result of a paper by Claude Shannon
- Recognized that a circuit is a rule for a function
  - Boolean Function
- Hardware was expensive
  - Repairs cost both money and time
- Talented Mathematicians & Engineers fully exploited Boolean Algebra to prevent defects.
- Theorems showed it possible to build any function from NAND gates
  - Highly economical approach

[1]Based on the paper “A Tale of Three Disciplines… and a Revolution” by Jesse H. Poore

Circuit Engineering History

- Mathematicians produced normal forms for functions using specific operators
- Notation lead to exact correspondence between diagrams and circuits
- Engineers designed larger units mathematically & constructed them physically
- Units evolved into components and reuse was born.
  - Based on exact functional descriptions of the unit
  - Streamlined innovation, based on catalogs of components
- By 1980’s complexity had increased state space enormously, yet could be treated explicitly

[1]Based on the paper “A Tale of Three Disciplines… and a Revolution” by Jesse H. Poore

Circuit Engineering Today

- Adheres to a description of components assembled according to the mathematics of functional composition
- Is still mostly a matter of forward design using established components and leaving a documented design trail.
- The complexity of circuit engineering matches or exceeds that of any other field.

[1]Based on the paper “A Tale of Three Disciplines… and a Revolution” by Jesse H. Poore

Conclusion

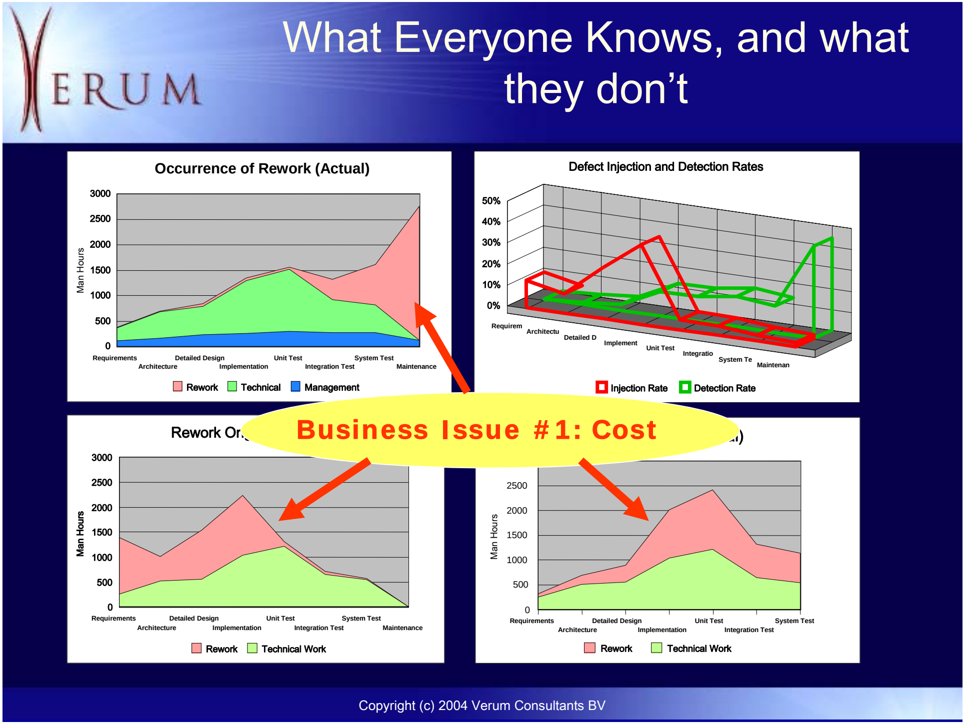
- In the field of Engineering most closely related to Software Engineering
- Mathematics has grown up hand-in-hand with design and implementation from the very earliest days
- As a result Circuit Engineers are able to manage the complexity of sophisticated components
- They are able to design and verify the designs of these components
- And the results of Circuit Engineering are highly predictable in comparison with its complexity.

Contents

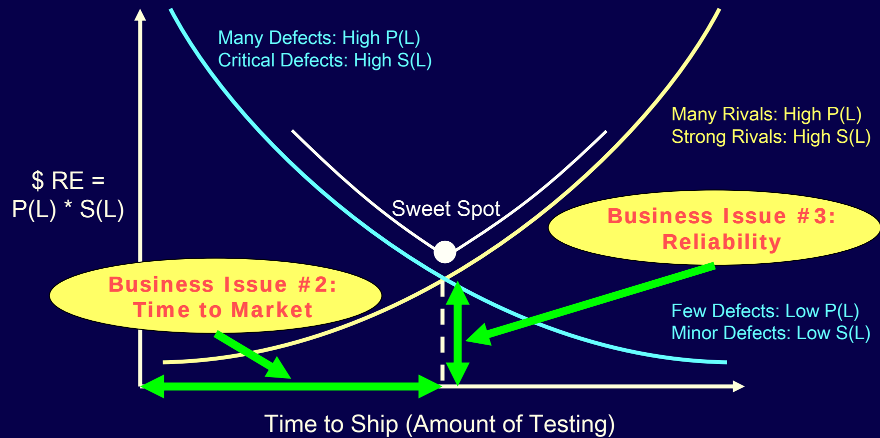
Studies and Reports

- Software Errors cost US economy $59.5 Billion in 2002
  - [NIST: The Economic Impacts of Inadequate Infrastructure for Software Testing]
- About 40-50% of the effort on current software projects is spent on avoidable rework
  - [DACS: A Business Case for Software Process Improvement Revised]
- About 40-50% of user programs enter use with nontrivial defects.
  - [CeBASE: Software Defect Reduction Top-10 List]
- 15% - 25% of all software defects are delivered to customers
  - [DACS]

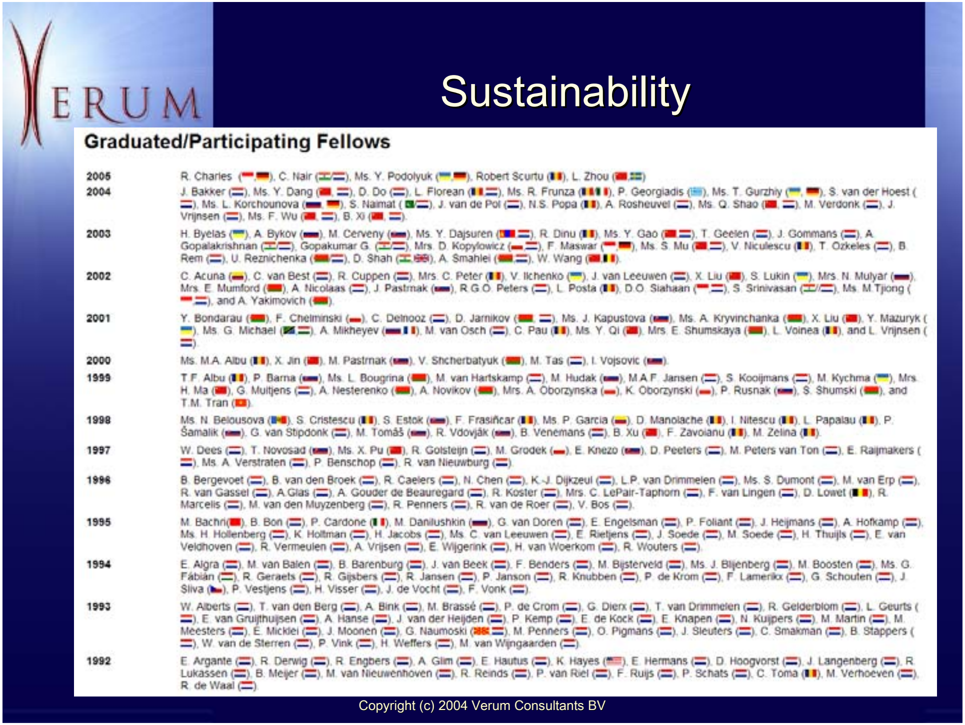

Planning Report 02-3
The Economic Impacts of Inadequate Infrastructure for Software Testing
Prepared by: RTI for
National Institute of Standards & Technology
Program Office Strategic Planning and Economic Analysis Group
May 2002
NIST


ERUM
What Everyone Knows, and what they don't
Occurrence of Rework (Actual)
Man Hours
3000
2500
2000
1500
1000
500
0
Requirements
Architecture
Detailed Design
Implementation
Unit Test
Integration Test
System Test
Maintenance
Rework
Technical
Management
Defect Injection and Detection Rates
50%
40%
30%
20%
10%
0%
Requirem
Architectu
Detailed D
Implement
Unit Test
Integratio
System Te
Maintenan
Injection Rate
Detection Rate
Business Issue #1: Cost
Rework Or
Man Hours
3000
2500
2000
1500
1000
500
0
Requirements
Architecture
Detailed Design
Implementation
Unit Test
Integration Test
System Test
Maintenance
Rework
Technical Work
Man Hours
2500
2000
1500
1000
500
0
Requirements
Architecture
Detailed Design
Implementation
Unit Test
Integration Test
System Test
Rework
Technical Work
Copyright (c) 2004 Verum Consultants BV

Release Economics



Source: B. Boehm, Centre for Software Engineering, USC


VERUM
Sustainability
Graduated/Participating Fellows
2005 R. Charles, C. Nair, Ms. Y. Podolyuk, Robert Scurtu, L. Zhou
2004 J. Bakker, Ms. Y. Dang, D. Do, L. Florean, Ms. R. Frunza, P. Georgiadis, Ms. T. Gurzhiy, S. van der Hoest, Ms. L. Korchounova, S. Naimat, J. van de Pol, N.S. Popa, A. Rosheuvel, Ms. Q. Shao, M. Verdonk, J. Vrijnsen, Ms. F. Wu, B. Xi.
2003 H. Byelas, A. Bykov, M. Cerveny, Ms. Y. Dajsuren, R. Dinu, Ms. Y. Gao, T. Geelen, J. Gommans, A. Gopalakrishnan, Gopakumar G., Mrs. D. Kopylowicz, F. Maswar, Ms. S. Mu, V. Niculescu, T. Ozkeles, B. Rem, U. Reznichenka, D. Shah, A. Smahlei, W. Wang.
2002 C. Acuna, C. van Best, R. Cuppen, Mrs. C. Peter, V. Ilchenko, J. van Leeuwen, X. Liu, S. Lukin, Mrs. N. Mulyar, Mrs. E. Mumford, A. Nicolaas, J. Pastrnak, R.G.O. Peters, L. Posta, D.O. Siahaan, S. Srinivasan, Ms. M. Tjiong, and A. Yakimovich.
2001 Y. Bondarau, F. Chelminski, C. Delnooz, D. Jarnikov, Ms. J. Kapustova, Ms. A. Kryvinchanka, X. Liu, Y. Mazuryk, Ms. G. Michael, A. Mikheyev, M. van Osch, C. Pau, Ms. Y. Qi, Mrs. E. Shumskaya, L. Voinea, and L. Vrijnsen.
2000 Ms. M.A. Albu, X. Jin, M. Pastrnak, V. Shcherbatyuk, M. Tas, I. Vojsovic.
1999 T.F. Albu, P. Barna, Ms. L. Bougrina, M. van Hartskamp, M. Hudak, M.A.F. Jansen, S. Kooijmans, M. Kychma, Mrs. H. Ma, G. Muitjens, A. Nesterenko, A. Novikov, Mrs. A. Oborzynska, K. Oborzynski, P. Rusnak, S. Shumski, and T.M. Tran.
1998 Ms. N. Belousova, S. Cristescu, S. Estok, F. Frasiñcar, Ms. P. Garcia, D. Manolache, I. Nitescu, L. Papalau, P. Šamalik, G. van Stipdonk, M. Tomáš, R. Vdovják, B. Venemans, B. Xu, F. Zavoianu, M. Zelina.
1997 W. Dees, T. Novosad, Ms. X. Pu, R. Golsteijn, M. Grodek, E. Knezo, D. Peeters, M. Peters van Ton, E. Raijmakers, Ms. A. Verstraten, P. Benschop, R. van Nieuwburg.
1996 B. Bergevoet, B. van den Broek, R. Caelers, N. Chen, K.-J. Dijkzeul, L.P. van Drimmelen, Ms. S. Dumont, M. van Erp, R. van Gassel, A. Glas, A. Gouder de Beauregard, R. Koster, Mrs. C. LePair-Taphorn, F. van Lingen, D. Lowet, R. Marcelis, M. van den Muyzenberg, R. Penners, R. van de Roer, V. Bos.
1995 M. Bachri, B. Bon, P. Cardone, M. Danilushkin, G. van Doren, E. Engelsman, P. Foliant, J. Heijmans, A. Hofkamp, Ms. H. Hollenberg, K. Holtman, H. Jacobs, Ms. C. van Leeuwen, E. Rietjens, J. Soede, M. Soede, H. Thuijls, E. van Veldhoven, R. Vermeulen, A. Vrijsen, E. Wijgerink, H. van Woerkom, R. Wouters.
1994 E. Algra, M. van Balen, B. Barenburg, J. van Beek, F. Benders, M. Bijsterveld, Ms. J. Blijenberg, M. Boosten, Ms. G. Fábián, R. Geraets, R. Gijsbers, R. Jansen, P. Janson, R. Knubben, P. de Krom, F. Lamerikx, G. Schouten, J. Sliva, P. Vestjens, H. Visser, J. de Vocht, F. Vonk.
1993 W. Alberts, T. van den Berg, A. Bink, M. Brassé, P. de Crom, G. Dierx, T. van Drimmelen, R. Gelderblom, L. Geurts, E. van Gruijthuijsen, A. Hanse, J. van der Heijden, P. Kemp, E. de Kock, E. Knapen, N. Kuijpers, M. Martin, M. Meesters, E. Micklei, J. Moonen, G. Naumoski, M. Penners, O. Pigmans, J. Sleuters, C. Smakman, B. Stappers, W. van de Sterren, P. Vink, H. Weffers, M. van Wijngaarden.
1992 E. Argante, R. Derwig, R. Engbers, A. Glim, E. Hautus, K. Hayes, E. Hermans, D. Hoogvorst, J. Langenberg, R. Lukassen, B. Meijer, M. van Nieuwenhoven, R. Reinds, P. van Riel, F. Ruijs, P. Schats, C. Toma, M. Verhoeven, R. de Waal.
Copyright (c) 2004 Verum Consultants BV

Conclusions

- Principle #1
  - Business Critical Software must be based on designs that are verifiably correct before implementation
- Principle #2
  - Software Architects and Designers must limit themselves to designs that can be verified
- Principle #1
  - The new Airbus A380 must be based on designs that are verifiably correct before implementation
- Principle #2
  - Aeronautical Engineers must limit themselves to designs that can be verified

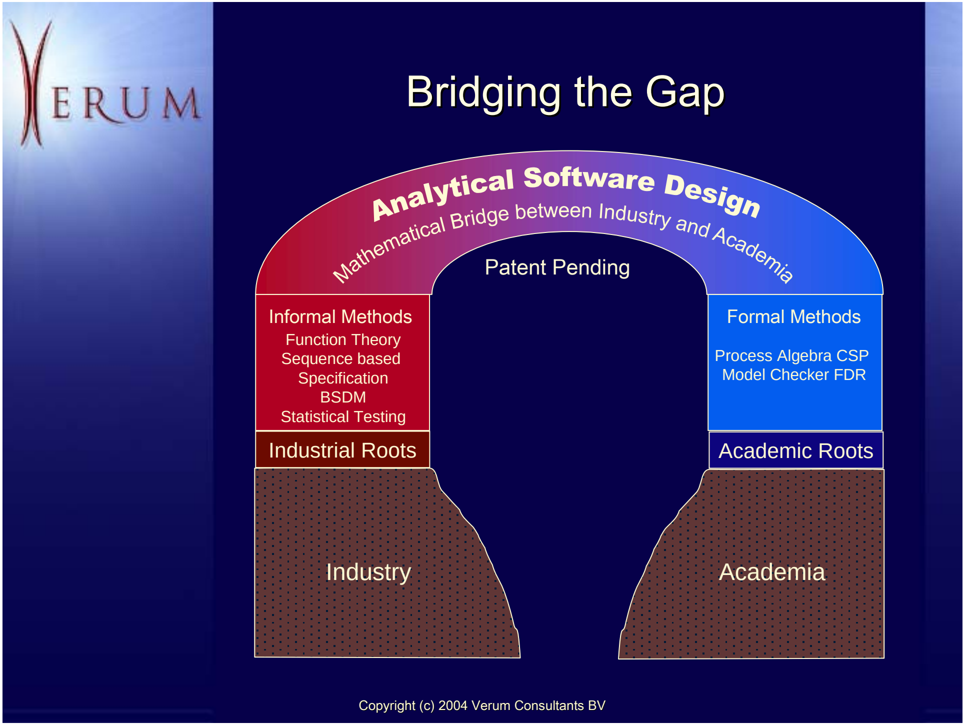
Contents

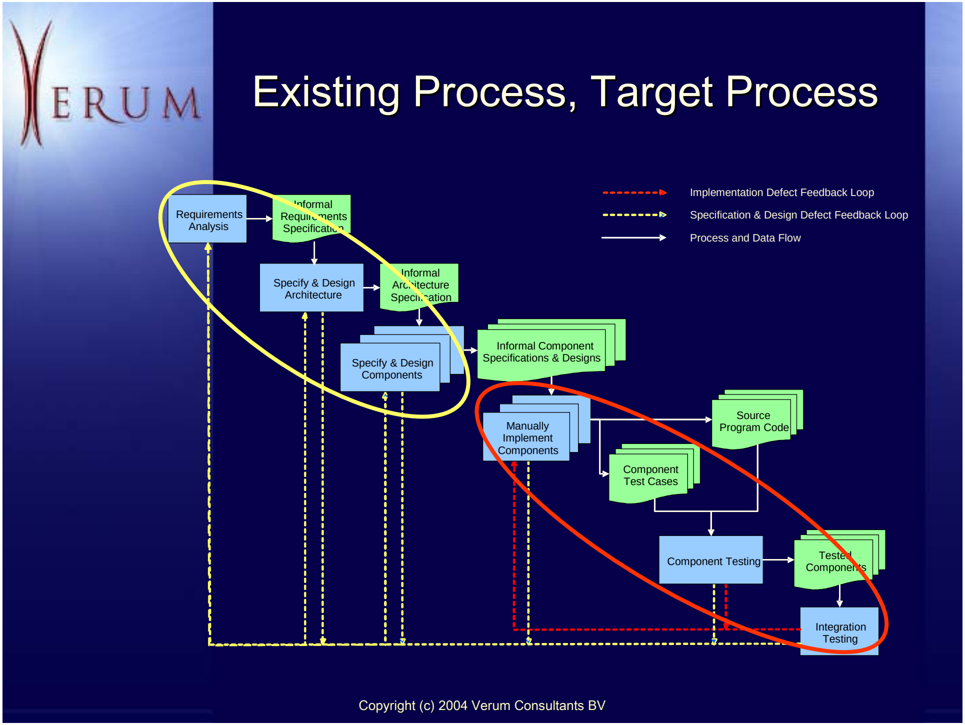
Issue: Predictability

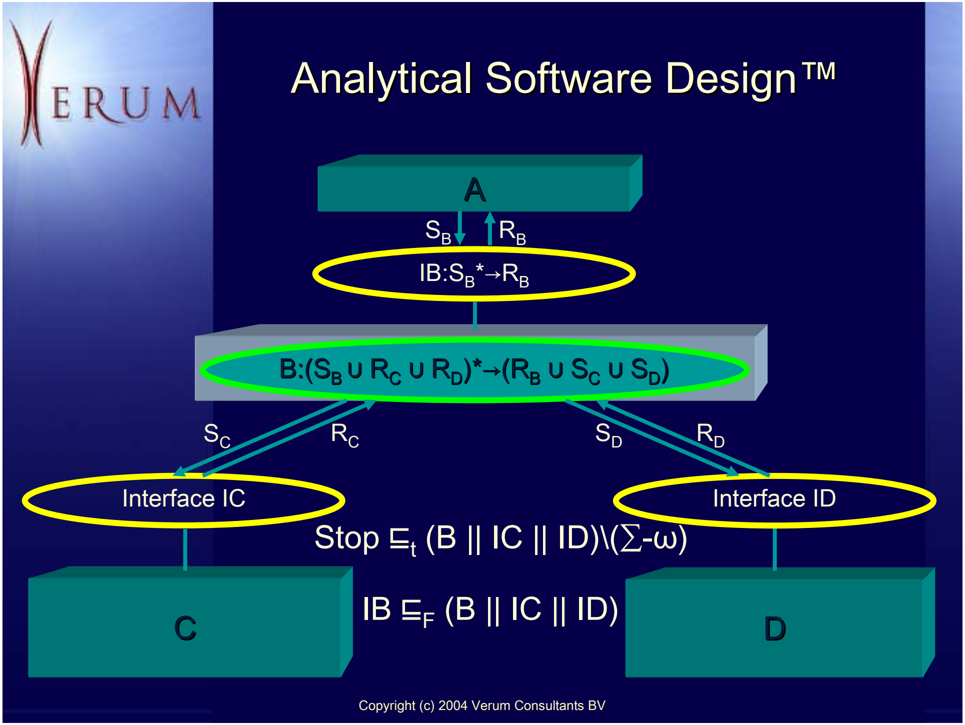
- How much will it cost ?
- When will it be ready ?
- How well will it work ?

Solution Domain

- Business Critical Software
  - Essential component of some core product or service
  - Software enables Competitive Advantage
  - Software poses Business Threat
  - Software dictates Time to Market
  - Software consumes disproportionate resources
- Embedded and In-Product Markets
  - OEM, Automotive, Medical, Telecoms, Process Control
- Systems with Complex Behaviour
  - Control Software, State Machine and Concurrent Applications
- Focus on Architecture, Design and Verification


VERUM
Bridging the Gap
Analytical Software Design
Mathematical Bridge between Industry and Academia
Patent Pending
Informal Methods
Function Theory
Sequence based Specification
BSDM
Statistical Testing
Formal Methods
Process Algebra CSP
Model Checker FDR
Industrial Roots
Academic Roots
Industry
Academia
Copyright (c) 2004 Verum Consultants BV

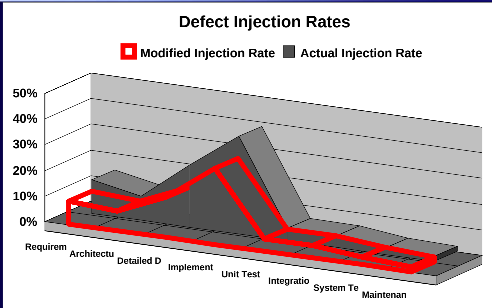

Existing Process, Target Process
Implementation Defect Feedback Loop
Specification & Design Defect Feedback Loop
Process and Data Flow
Requirements Analysis
Informal Requirements Specification
Specify & Design Architecture
Informal Architecture Specification
Specify & Design Components
Informal Component Specifications & Designs
Manually Implement Components
Source Program Code
Component Test Cases
Component Testing
Tested Components
Integration Testing
Copyright (c) 2004 Verum Consultants BV

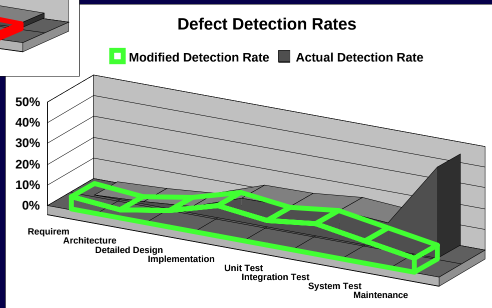

VERUM
Analytical Software Design™
A
S_B
R_B
IB:S_B*→R_B
B:(S_B ∪ R_C ∪ R_D)*→(R_B ∪ S_C ∪ S_D)
S_C
R_C
S_D
R_D
Interface IC
Interface ID
Stop ⊑_t (B || IC || ID)\(Σ-ω)
IB ⊑_F (B || IC || ID)
C
D
Copyright (c) 2004 Verum Consultants BV

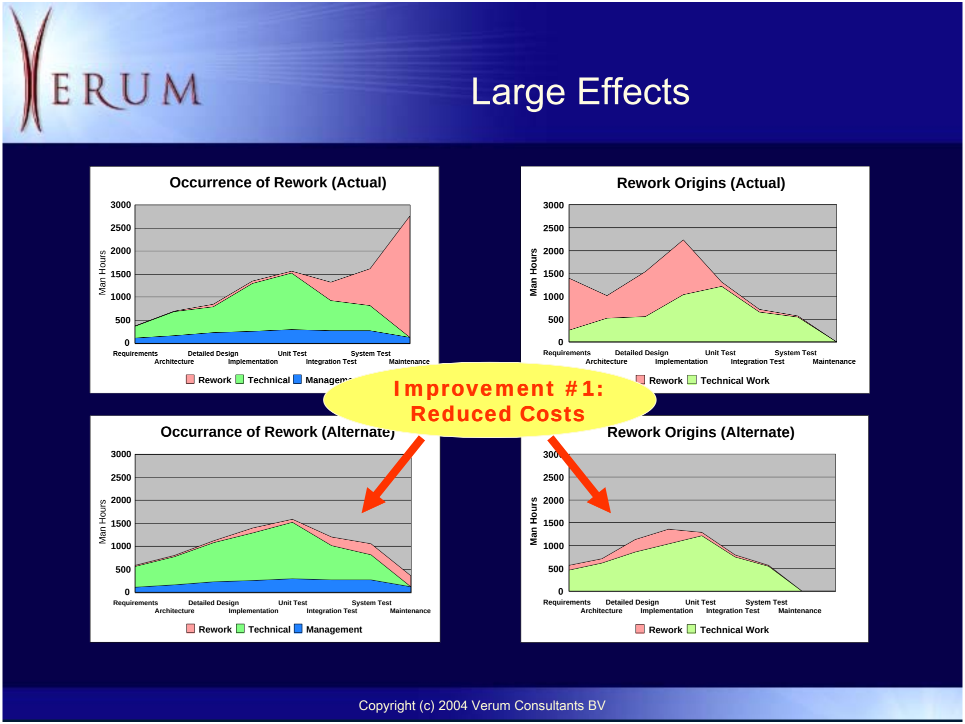
VERUM

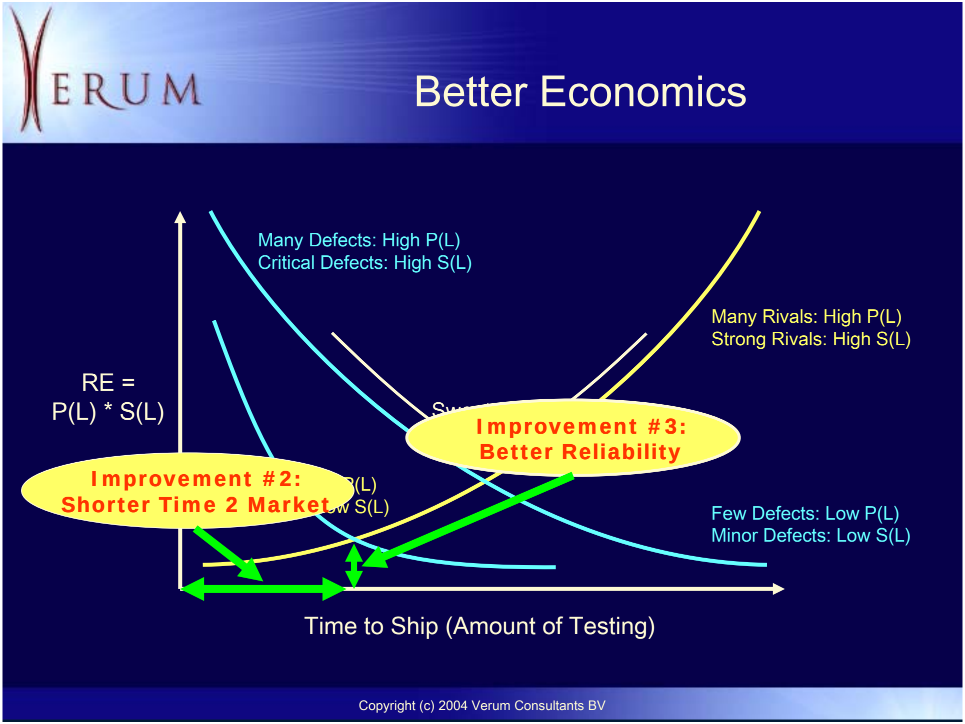
Contents

Small Changes






VERUM
Large Effects
Occurrence of Rework (Actual)
Rework Origins (Actual)
Occurrance of Rework (Alternate)
Rework Origins (Alternate)
Man Hours
3000
2500
2000
1500
1000
500
0
Requirements
Architecture
Detailed Design
Implementation
Unit Test
Integration Test
System Test
Maintenance
Rework
Technical
Management
Technical Work
Improvement #1:
Reduced Costs
Copyright (c) 2004 Verum Consultants BV


ERUM
Better Economics
Many Defects: High P(L)
Critical Defects: High S(L)
Many Rivals: High P(L)
Strong Rivals: High S(L)
RE =
P(L) * S(L)
Improvement #3:
Better Reliability
Improvement #2:
Shorter Time 2 Market
Few Defects: Low P(L)
Minor Defects: Low S(L)
Time to Ship (Amount of Testing)
Copyright (c) 2004 Verum Consultants BV

Contents

Making Software Fly

- Software Engineering must:
  - adopt methods and techniques that are routine for every other branch of engineering
  - must embrace the use of mathematics at all levels of education & industry
- Specifically:
  - Business Critical Software must be based on designs that are verifiably correct before implementation
    Software Architects and Designers must limit themselves to designs that can be verified
- These things are now economically possible